

SOUTHWEST RESEARCH INSTITUTE
Post Office Drawer 28510, 6220 Culebra Road
San Antonio, Texas 78228-0510

CONTINUATION OF RESEARCH IN SOFTWARE FOR SPACE OPERATIONS SUPPORT

SEMIANNUAL REPORT

NASA Grant No. NAG 9-388
SwRI Project No. 05-2984

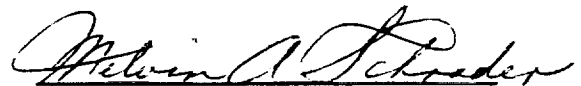
JOHNSON
GRANT
IN-61-CR
252150
79P

Prepared by:
Mark D. Collier

Prepared for:
NASA
Johnson Space Center
Houston TX 77058

January 4, 1989

Approved:



Melvin A. Schrader, Director
Data Systems Science and
Technology Department

(NASA-CR-186092) CONTINUATION OF RESEARCH
IN SOFTWARE FOR SPACE OPERATIONS SUPPORT
Semiannual Report (Southwest Research
Inst.) 79 p

N90-14808

CSCL 09P

Unclass

63/61 0252150

Table of Contents

1.0 Introduction	1
2.0 Research Background	1
2.1 NASA Grant NAG 9-269 Background	1
3.0 Research Direction	2
3.1 Phase 1 Progress	2
3.1.1 Workstation Executive Release 2.5	3
3.1.2 Review of Graphics Standards.....	3
3.1.3 POSIX Real-time and Security Extensions (1003.4 and 1003.6).....	3
3.1.4 MOTIF User Interface Standard	3
4.0 Future Research Directions.....	4
Appendix - A.....	5
Appendix - B	6
Appendix - C	7
Appendix - D.....	8
Appendix - E	9

1.0 Introduction

This document serves as the semiannual report describing the activity on NASA Grant NAG 9-388, which is entitled "Continuation of Research in Software for Space Operations Support". The purpose of this grant is to continue the research direction defined for NASA Grant NAG 9-269, during which SwRI developed a prototype workstation executive called the Hardware Independent Software Development Environment (HISDE). The research direction of this grant is to continue to research and evaluate software technologies relevant to workstation executives and to use HISDE as a test bed for prototyping efforts.

This document will describe the background for the research grant, define all research directions, and summarize progress to date. Included are appendices for documents generated thus far.

2.0 Research Background

During the past few years and increasing in the future, many centralized computing installations are migrating to environments characterized by distributed processing. This migration is driven primarily by the low cost and high performance delivered by state-of-the-art graphic workstations. Such an environment normally consists of a large number of workstations which are in turn connected via one or more high-speed networks.

Although a workstation-based distributed processing environment offers many advantages, it also introduces a number of new concerns. One problem is that engineering-class workstations most commonly use the UNIX operating system, which is difficult for computer novices to use effectively. Also, connecting a large number of workstations and expecting them to work as an integrated system is not easily achieved. The introduction of so many separate processors makes configuration management and security a real concern. In fact, the very flexibility which is inherent in workstations often becomes a problem. This is especially true for real-time critical command and control systems in which a failure or security break could have disastrous results.

In order to solve these problems, allow the environment to function as an integrated system, and present a functional development environment to application programmers, it is necessary to develop an additional layer of software. This "executive" software integrates the system, provides real-time capabilities, and provides the tools necessary to support the application requirements. Such an executive will be required for use in evolving systems such as the ground-based control centers planned at Johnson Space Center. These command and control environments will use a distributed processing architecture to provide real-time processing of telemetry and command data.

2.1 NASA Grant NAG 9-269 Background

For NASA Grant NAG 9-269, which was entitled "Research in Software for Space Operations Support", SwRI developed the HISDE prototype to serve as proof-of-concept for a hardware-independent workstation executive. The HISDE prototype introduced a number of advanced software technologies and concepts including:

- Exclusive use of software standards:
 - SVID UNIX Operating System
 - X Windows

Semiannual Status Report

- GKS and PHIGS
- ISO OSI Communications

Through the use of standards, HISDE was easily ported across multiple vendor workstations.

- Open use of UNIX - through a more flexible design and configuration management scheme, HISDE provided access to the UNIX file system via the familiar UNIX command line interface.
- CM Manager Workstation - this concept involves a workstation to which all user applications are loaded with certified libraries prior to being mission certified and uploaded to configuration management host.

The purpose of this continuation grant is to research software technologies relevant to workstation executives. The HISDE prototype will be used as a test bed for prototyping and practical evaluation of identified technologies.

3.0 Research Direction

The research direction for this grant is divided into two independent, on-going phases. The phases and the included research are as follows:

- Phase 1 - Research, evaluate, and prototype identified software technologies. At this time, efforts have been devoted to:
 - Workstation Executive Release 2.5
 - Graphics standards
 - POSIX Real-time and Security Extensions (1003.4 and 1003.6)
 - MOTIF user interface standard
- Phase 2 - In cooperation with activity on NAG 9-340, use HISDE as a test bed for identified prototyping efforts.

This grant is proceeding in cooperation with NASA Grant NAG 9-340, which is entitled "Research into Software Executives for Space Operations Support". The focus of NAG 9-340 is to survey industry for technology relevant to software executives, define the functionality in an executive, develop a set of requirements for a concept executive, and then apply the concept executive to an actual NASA environment by identifying and developing prototypes. Such prototypes will be demonstrated within or upgrade the existing HISDE environment. The timing of NAG 9-340 is such that the concept executive will be presented in early February, 1990. As such, the individual prototyping efforts have not yet been defined.

The following two sections will describe in more detail the research progress of the first two phases. Phase 2 is not described as no research has yet been performed.

3.1 Phase 1 Progress

The purpose of this phase is to research and review identified software technologies which are relevant to workstation executives. At this time, the following distinct research efforts have taken place:

- Workstation Executive Release 2.5
- Graphics standards

- POSIX real-time and security extensions (1003.4 and 1003.6)
- MOTIF user interface standard

The research direction taken for each topic is described in more detail in the following subsections.

3.1.1 Workstation Executive Release 2.5

The Workstation Executive (WEX) Release 2.5 is a subsystem currently under development at NASA Johnson Space Center. In order to remain up to date with the current direction of workstation executives in this environment, SwRI evaluated the design for this subsystem. During the evaluation, SwRI generated comments concerning the design of this subsystem. The generated documents are found in Appendices A and B.

3.1.2 Review of Graphics Standards

SwRI researched and generated a brief report summarizing effective use of graphics standards in relation to the Mission Control Center Upgrade (MCCU) environment. This report describes graphics software which is currently available, how the software applies to the MCCU environment, and describes which software is useful for certain requirements. The generated document is found in Appendix C.

3.1.3 POSIX Real-time and Security Extensions (1003.4 and 1003.6)

The Portable Operating System Interface Definition (POSIX) standard 1003.1 defined by IEEE is projected as the de-facto standard for operating systems. As such it is of interest to the research performed for workstation executives. At this time, POSIX only specifies the programmatic interface to the basic operating system (system calls). However, a number of working groups are developing interfaces for required extensions, such as command line interaction, test verification, real-time extensions, ADA function bindings, security, system administration, and networking.

NASA-JSC is in the process of generating an Operating System Interface Definition (OSID) which defines the functional interface requirements for operating systems to be procured. NASA-JSC is attempting to review the POSIX working drafts to determine if all JSC requirements will be satisfied for operating system interfaces. SwRI aided this effort by evaluating the POSIX real-time and security extensions drafts for this purpose.

For the POSIX real-time extensions (POSIX 1003.4), SwRI generated OSID work sheets which indicate functional or interface problems in the working draft. For POSIX security extensions (POSIX 1003.6), SwRI examined the JSC Automated Information Security Plan (AIS) document to determine if the described requirements were met by the working draft. The generated documents are found in Appendices E and F.

3.1.4 MOTIF User Interface Standard

MOTIF is a user interface system developed by the Open Software Foundation (OSF). MOTIF is based upon the X Windows and Xt intrinsics standards and provides a complete user interface development system. This includes the following libraries and applications:

- A widget set - a 3-dimensional widget set which is widely acknowledged as the most complete in the industry.

- A window manager - a window manager which implements IBM/Microsoft's "presentation manager" behavior. This behavior defines a consistent look and feel of the user interface.
- A user interface language (UIL) compiler - an application which allows a programmer to develop a user interface without writing C code.

MOTIF is widely supported (or support is planned) by the majority of workstation vendors. As such, MOTIF is a candidate for selection as the user interface software used by a workstation executive. To fully evaluate the MOTIF system, SwRI performed the following:

- procured a MOTIF source license from OSF,
- installed/porting MOTIF to Sun and MASSCOMP architectures,
- ported the HISDE prototype to MOTIF, and
- demonstrated the ported HISDE prototype to NASA personnel.

At this time no formal review of MOTIF has been generated, as there are a number of outstanding questions concerning use of the UIL, MOTIF executable size, and performance in general. SwRI plans to use MOTIF as the basis for the user interface for subsequently developed prototypes. During this effort, SwRI will find answers to the remaining questions and generate a complete report.

4.0 Future Research Directions

The future direction of this grant will depend on the prototyping efforts identified as part of NAG 9-340. The specific prototypes to be developed will most likely include a subset of the following:

- Configuration independence - an executive design which supports various configurations of workstation hardware:
 - Tightly coupled architecture with integrated graphics processors
 - Diskless/Dataless client workstations
 - X terminals
- Intelligent process loading - an executive design which allows CPU-intensive processes to be automatically executed on the workstation with the lowest processing load.
- User interface prototyping in MOTIF - an executive which uses MOTIF and other high-level user interface software for development of all user interfaces.
- POSIX - an executive design which uses POSIX interfaces for operating system and real-time interaction.

It is also likely that additional topics relevant to workstation executives will be identified and researched.

**Appendix - A
Comments on
Preliminary Design Review Document**

Comments on WEX 2.5 Preliminary Design

1.0 Overview

This document describes any problems (potential or otherwise) seen with IBM's Workstation Executive (WEX) 2.5 preliminary design. The PDR document was divided into 12 sections and several appendices. This document includes comments on the sections describing workstation software. This includes the following:

<u>Section #</u>	<u>Section Title</u>
1	System Overview
2	GPLAN/RTLAN
3	WEX
7	Data Acquisition
9	Configuration Management
11	2.5 PDR Systems Analysis

Note that no comments were generated for section 12 (CM Workstation), as this is a prototype for delivery 2.5. The described design appears to be sound, however there are details which must be completed before this application can provide operational support.

The following sections of this document include general comments and then specific comments for each applicable section. For sections providing specific comments, the page in the PDR document is given and then is followed by the appropriate discussion.

General Comments

In general, the design presented by IBM is sound. Many of the concepts demonstrated by the Hardware Independent Software Development Environment (HISDE) prototype were reflected in the 2.5 design. From the 2.5 PDR, it is obvious that IBM is migrating towards a hardware independent design. This is evident in the use of X Windows as the primary graphics software. In recent versions of WEX, one of the primary problems is the dependence on proprietary graphics systems (namely MGI graphics). By removing dependence on this software and forcing users to do the same is a major step toward writing portable code, as the majority of non-standard user software is in the area of graphics use.

The 2.5 design also includes use of ISO standards as implemented by the RETIX corporation. These ISO standards are based on the Open Systems Interconnection (OSI) reference model and is an important step toward standard usage and communication over LANs. However, it is important for IBM to recognize that such a move will initially cause problems due to existing software which is based on other standards (namely TCP/IP). TCP/IP is relatively fast, reliable, and has a wealth of application software which relies upon it. During the transition from TCP/IP to ISO, some means of retaining these benefits must be retained. Users and developers will not want to lose features such as network X Windows, NFS, EMAIL, and others, which currently work well over TCP/IP.

There is concern about IBM's understanding of what is SVID UNIX and how it will be verified that their software is truly compliant. IBM should not state that their software is compliant when it allows or depends on functions which are unique to Berkeley UNIX. IBM's design should allow the user of non-SVID commands and features, but should constantly remind users that such functions may not be available on other systems. Also, IBM's design should in no way depend on any non-SVID commands or functions.

The primary concern about the 2.5 design is that it is not completely hardware independent. Although IBM states that the design is in fact hardware independent, there is no provision whereby this will be proven (such as porting software to another workstation system). In addition, the basic design is still dependent on a tightly-coupled processing configuration, in which there is no distribution of data or processing. All computation (less that on the MASSCOMP IGP's) and data is resident on the main CPU(s). This design is not flexible enough to support a true diskless node configuration, in which each node has a significant amount of local processing power. The diskless node approach is one chosen by many of the industry workstation vendors and cannot be ignored if one is attempting to produce a hardware, vendor, and configuration independent system. This is a difficult problem and one in which there are many opinions but only a few reasonable solutions.

2.0 System Overview Comments

Page 1-10

The document states that the 2.5 design is hardware independent. Again, this is not true, as the design is neither vendor nor configuration independent. If the design is truly hardware independent, then some plan to prove this should be devised.

3.0 GPLAN/RTLAN Comments

Page 2-4

There is no reference to how existing TCP/IP-based services will be provided in the interim before they are supported via ISO mechanisms. This will preclude use of services such as network X Windows, NFS, UNIX EMAIL, etc.

Page 2-5

There is no indication of how many of the ISO layers are implemented in the UNIX kernel. From previous discussions, layers 3 - 7 operate at the user level. This will may make communications extremely slow. IBM should be requested to provide benchmarks which compare movement of data through existing and then the ISO communications routes. Note that it is unlikely that ISO will ever be faster than existing methods, but there are steps that can be taken to improve performance (such as placing more ISO software into the UNIX kernel).

Page 2-14

Why is it necessary for the user to format data into 8K blocks? While it is understandable to maintain this functionality to be compatible with WEX 2.3, why not provide a new utility which allows access in a more UNIX-like fashion?

Page 2-15

Is it true that LAN services do not require WEX shared memory? The shared memory segment allocated by WEX is quite large (about 0.5 to 1.0 MEG). It is unreasonable for a user to lose resources to WEX shared memory if only LAN access is required. Separating the two services is an excellent design idea, but care should be taken to avoid any dependencies between the two software systems (such as LAN services requiring WEX shared memory services to determine flight or operation mode).

Page 2-26

What is the purpose of "network parameters"? If these are upper limits, what will happen to data which exceeds the limits?

Page 2-31

Is the data saved by the network manager available for local display to the user? Also, is there some sort of client which allows this data to be displayed in an interactive and meaningful manner? Finally, can this data be routed to the Health and Status application?

Page 2-32

Will the file transfer and access capabilities of FTAM allow use of wildcards and recursive access of directories? Users will require the ability to copy the contents of entire directories and hierarchies. This capability will also be required for WEX applications which move files to and from the host and other workstations. If FTAM itself does not provide these capabilities, a set of shell functions must be provided. These functions must be available for interactive and programmatic usage.

There is no mention of how IBM's FTAM will honor normal UNIX file ownership. From experience, FTAM does not do an adequate job with file ownership. The ability to entirely enable/disable workstation access is insufficient. Users will require permissions to be recognized on a per file basis (just like existing UNIX services provide):

Page 2-37

FTAM's honoring of the UNIX "other" ownership is inadequate. This precludes recognition of user and group ownership.

Marking a file as "non-executable" is a good idea, but is by no means fail-safe. If the user is able to write the file, he will be able to change the permissions on it, thus quickly making it executable. If FTAM is the sole means of copying files over the network, it could be used to determine if a file is truly executable (via examining magic numbers) and if so, prevent it from being copied. This would be a reasonable location to place such security checks.

Page 2-43

The network test tools should be available for programmatic access. This would allow an application to query the host and/or workstation before access.

4.0 WEX Comments

Page 3-4

WEX does not provide "vendor independent" applications software. If this is a primary goal, IBM should modify their basic design to support other vendors and configurations.

IBM states that the user may use alternate shells, window managers, terminal emulators, and other critical tools. This is acceptable during development mode, but during operational mode, such applications must not be allowed unless they are certified. It also makes sense to only allow use of standard applications during operational mode. This would further force users to develop portable code and not depend on non-standard tools.

Page 3-5

WEX does not adhere to SVID UNIX. It depends on and allows Berkeley-specific tools (for example, the document describes two startup sequences, one of which is unique to Berkeley UNIX). It also does not specify any means whereby SVID compliance is to be verified.

ANSI C extensions should not be used until they become more universally accepted by workstation vendors.

Page 3-6

Stating that WEX applications "will run with or without the window manager" involves a significant amount of functionality. This means that if a window is resized (larger or smaller), the client will intelligently resize areas and fonts. This is not an automatic process (at least not "intelligently"), as most "widgets" do a poor job of handling resize operations.

Page 3-7

WEX must not preclude the use of PHIGS. The current architecture, if used with true X terminals or diskless nodes (treated as X terminals), may preclude use of PHIGS if the software bypasses the X server. At the current time, much high-performance graphics software bypasses the X server. The graphics software cooperates with the window system, but does not route display requests through the X server. Instead, the underlying graphics libraries are directly accessed. This is primarily due to lack of X protocol features for functions such as 3-dimensional rendering. In the future, the X protocol will be expanded to support such functionality, but in the interim, such graphics will not use the X server and therefore are not network transparent.

Mouseless mode is an interesting problem which may be solvable. It is definitely solvable with access to server source code.

Page 3-8

WEX should provide a user interface language (UIL). This includes some sort of high-level language for definition and separation of user interfaces. This language is accessed programmatically and/or through an interactive display builder (such as that provided by TAE). Note that the Display Builder application (as defined by FAC), is not general purpose enough to support this requirement. A UIL would drastically reduce the amount of time required to develop WEX and application user interfaces. Also, as the user interface specifics are separated from the actual applications, it makes it very easy to alter the interface.

Page 3-12

WEX does not support diskless nodes as separate processors. It only allows them to be used as dedicated X Windows terminals. This would waste the local processing capability of the diskless nodes.

WEX will not be capable of supporting X terminals. Such devices have minimal graphics support and rely heavily on the host (the MASSCOMP) for computation. Using X terminals would place an unreasonable burden on the MASSCOMP (a burden which is currently relieved by the dedicated graphics processors in the IGP's).

Page 3-14

If WEX is only going to run on the MASSCOMP 6600's and is SVID compliant, then why are two start-up scenarios presented (one of which is not SVID)?

Page 3-18

Why is WEX shared memory allocated even if the user is not planning to use any WEX service?

Page 3-19

The WEXTASKS file requires a new language. As an alternative, use standard UNIX services to provide the required functionality.

Page 3-22

Is it possible for the user to send an advisory to a user on any given workstation? The document states that an advisory may be sent to any tty, but not another workstation.

Page 3-23

If a user is logging into the system in OPERATIONAL mode, where is the login verification data coming from? Is it the local CM process or from the host? This is most likely from the host, but if local, some sort of extra security is required to prevent unauthorized access to the data.

Page 3-25

In the "xterm" window provided during login, it is not clear what application is running (it is normally the shell). This is of course not reasonable during login. Also, the xterm window should have a scrollbar to allow the user to review messages which have scrolled off the top of the screen.

Page 3-26

The document references "keyboard-to-button" mappings, which are assumed to be inputs such as tab and cursor keys. Such mappings **MUST** be consistent from WEX application to application. These mappings must be sufficient to allow a user to complete a screen without ANY mouse interaction. It is very frustrating to be forced to use a combination of a mouse and keyboard interface.

The keyboard mappings must also be provided to users so that they may make their own applications consistent with those in WEX.

Visually disabling buttons or fields is in many cases a costly and unnecessary action. It is normally adequate to use a simple mechanism such as preventing the cursor from changing when it tracks over a button. The best solution will depend on the user interface tools used (which widgets) and how efficiently they handle such requests.

Will the user be allowed to selectively kill background processes, or will it be a "all-or-nothing" action?

Starting up the users environment in the xterm window will be a problem, as some users will not want to present an xterm window to their flight controllers. If users want an xterm window available, have them specify it in the "xstuff" start-up file. Give the user a completely blank starting point from which they may set up the environment.

Page 3-30

If a download or recycle is requested, will users on other terminals be able to prevent it from occurring? If one user wants to enter a mode requiring such action, will it cause users in other modes to be logged off?

Page 3-31

Why are UNIX accounting files updated? Is this for security reasons? Using the UNIX accounting process will use a significant amount of resources.

Page 3-36

All common WEX default resources (colors, fonts, etc) should be stored in the server so that clients may be quickly initialized. Otherwise, the client will have to read and parse such data from a file each time it is executed. Note that storing defaults into the server is made possible with the "xrdb" client.

Page 3-41

Is the "icon" button on every title bar really necessary? If the reasoning is to account for situations in which no window manager is running, then buttons for move and resize should be considered. It is not a good design to have application-specific buttons for such common functions. This type of functionality is provided by other existing window managers. If IBM prefers a window manager which places move, resize, iconify, and other widgets on each window, then one providing these features should be selected and agreed upon.

Again, how intelligent will the resizing action be? On many clients, the user will only want to resize a portion of the window, such as that used for data entry or output (such as the message area of the advisory client). Any client which uses a large area for display of a variable amount of information should initialize separate windows (Xtoolkit shells). This allows the user to easily adjust the amount of text displayed.

Note that many simple messages (especially those requiring immediate response), are best processed with popup windows.

All important messages should be output to the advisory window if it is currently active. Note that local message windows are not useful if the advisory window is already present. They are only really useful if large amounts of data must be displayed (such large amounts of data would clog the advisory window).

Page 3-44

The document makes frequent reference to on-line help, but there is no sample screen nor any real definition of the functionality to be provided. Will it be context-sensitive (more ideal) or more of a window with "man" formatted data? An ideal solution would be to provide both types of help.

Is there a process which handles on-line help or is this a sub-function of all clients? It makes more sense to separate this function, as it will make clients easier to develop. It also will allow clients to run in parallel with the user obtaining help (without forcing the client to worry about retaining parallel operation). Note that this does make context-sensitive help more difficult to implement.

Will the local CM access functions be converted to X Windows? The windows shown appear to be in a "curses" type interface.

What type of display interface is being used by the applications which are not being converted to X Windows? WEX cannot claim a consistent "look and feel" if there are applications which use a different interface. Also, are these interfaces dumb screen I/O, curses, or

MASSCOMP graphics? All should be redone, but especially the MASSCOMP graphics, as this type of code is completely non-portable.

Page 3-47

As a general comment, on prompts which request entry of a number in a limited range, it is often easier to use a scrollbar to adjust the value. The best solution is to provide a text area in which data may be entered directly and some sort of widget which allows mouse-based manipulation. Using up/down buttons (as shown in the example) are inefficient for wide ranges of data in which precision is still required.

Page 3-48

How does the user acknowledge a message in the advisory client's window? What does the "browse" function do? It should provide additional functionality such as search for text, display on message type, sort, etc. The browse function itself should be integrated with the main window functions (make window bigger to browse).

Page 3-54

What does the "registration" process involve? This implies that all WEX shared memory is already allocated and the registration identifies the user to WEX, attaches the user to shared memory, and provides some resources (which the user may or may not need). It is not reasonable to allocate a large amount of shared memory unless the user uses the associated WEX services. Shared memory and other resources should be allocated dynamically as required by the WEX services used. If no WEX service is used, NO shared memory, message queues, semaphores, or other resources should be allocated. A typical user may only require data acquisition services and should not lose resources due to requirements of undesired services and applications.

Page 3-55

Most users sophisticated enough to use "message queue utilities", "interprocess communication utilities", and "shared memory buffer utilities" will neither require nor desire automatic initialization. Again, all WEX resources should be dynamically allocated and only when requested by the user. Why should a user lose a large amount of memory if he only needs to use a simple service (such as outputting an advisory message).

Page 3-56

What is the provision for returning resources after an application terminates unexpectedly (without calling EXwexterm)?

Page 3-62

Why is the "su" command provided? While a user may want to "su" to another user (other than root), it still implies that the users will have root access. If this is so, there cannot be any reasonable CM, no matter how stringent the policies are during operational mode (short of downloading the entire system).

Page 3-63

What is "proper security"? Is this a dependence on procedural control? Allowing a user to read in data allows import of executable files. It may be necessary to "front-end" any and all such commands.

There are a number of commands listed which could be dangerous during operational modes. For this scheme to work effectively, strict permissions must be maintained on all files.

5.0 Data Acquisition

Page 7-9

The user interface presented is inefficient in its use of display space. The areas for "Active MTM streams", "Active Positions", and pop-ups should not be static. It is better design to have separate windows which are popped-up as required (in the same manner described for local message windows).

Page 7-14

Why is the Uniform Network Interface not used for both LAN's (GP and RT)? Will the user be allowed to access the GP LAN via this set of tools?

Page 7-15

Is it guaranteed that the WEX shared memory is not required for LAN access? It appears that WEX shared memory is required to log in, so there is nothing gained by separating the resources used by the two services (other than one allocation being static while the other is dynamic).

6.0 Configuration Management

Page 9-6

Will the "Workstation Processor ID" adequately differentiate between different types of processors for a given vendor (MC5500 vs MC6600)?

An element will need to indicate for which release of the operating system it has been loaded for. It will also need to indicate the release of other major software, such as graphics, WEX, X, etc.

Page 9-12

How is the "integrity check" performed? What is the baseline against which data is compared? Is it simply update dates or some sort of checksum? Either way the baseline data must be resident on the host or protected from user access.

Is it possible for users to modify the permissions (make executable) on files in the operational area? If so, the user could easily make executable, a file which appears to be a normal binary data file.

Page 9-13

What does "linked into the operational area" imply? Does this mean (in the UNIX sense) that links are established to existing files somewhere else on the file system?

Are all executables downloaded or just those not already on the disk? If an integrity check is performed (and the check is dependable), is it necessary to download files which already exist? A complete download makes sense from a security standpoint, but will take a significant amount of time to complete.

Page 9-14

The described file tree indicates that the "chroot" command will be used. In such a scenario, many of the common areas (usr, bin, dev, etc) will have to be duplicated. Again, this makes sense from a security standpoint, but will waste disk for areas which are identical for all operational modes.

Page 9-26

The element information includes the "setuid" permission. Is it possible to set this bit and set application ownership to root, thus allowing the application to access the system as if it were superuser?

Page 9-38

How will "reads of executables" be prevented during operational mode? It appears that the "cpio" and "tar" commands will be available and they allow such operations.

Page 9-39

Is it likely that a user would ever initiate an integrity check? This might be used if the user is encountering a strange problem which he suspects is due to out-of-date code or system data. In such a case, a more useful feature might be a simple check which examines the revisions of OS, WEX, graphics, and applications.

Page 9-41

Again, how is an "uncertified" program found? How dependable is the method used to determine if a file is an executable? Is it simple permissions, magic numbers, or actual file content?

Page 9-44

This screen shown for CM does not indicate that it will be ported to X Windows. Does this mean that an existing curses (or something else) interface will be retained?

7.0 2.5 PDR Systems Analysis

Page 11-6

The cumulative MIPS figures indicate that the Flight Support Host will be used near its processing capacity. Expecting a system to have adequate performance and response after 75% load is unreasonable.

Page 11-20

Which release of X Windows was used for the benchmarks? MASSCOMP has released version 1.1, which is significantly faster than release 1.0 (Note that both are based on the MIT X11 Release 2 X Windows).

The table indicates that the Display Manager uses X Windows. Is this really the case, or will this application still use GKS as its primary graphics resource? Note that if it uses GKS, it may be slower than X Windows (depending on the type of fonts displayed).

Page 11-24

The document states that "6.4 MIPS" will be provided for user applications. This assumes that the machine is capable of efficiently providing 100% of its maximum rating processing capability. This is unreasonable, as when more and more processes are executed, an

Comments on WEX 2.5 Preliminary Design

increased amount of processing is lost to system overhead; therefore the "6.4 MIPS" number is optimistic.

Appendix - B
Comments on
Critical Design Review Document

1.0 Overview

This document describes any concerns (potential or otherwise) seen with applicable portions of the 2.5 critical design. This document concentrates on sections 5 and 6, which deal with WEX and local Configuration Management.

2.0 WEX Comments

Page 5-6

The term "vendor independent" is misleading. Although software is based on standards, it depends on a configuration which is quite unique within the workstation market (that being MASSCOMP's).

The document states that "WEX services will be provided on an optional basis". Is this absolutely true or do major applications like data acquisition and configuration management require initialization of WEX before they will operate properly?

Providing a "uniform look and feel" encompasses a broad area of user interface. For this statement to be accurate, a well defined standard for colors, menus, title bars, widget use and placement, keyboard mapping and other aspects must be defined. A functional standard for the user interface should be designed, presented in the form of a prototype, and used for all applications executing on the workstation. Note: will applications from other contractors have a different appearance? This will be a problem for users.

How significant is the requirement for ASCII terminal support? Will all WEX clients also provide ASCII (curses) interfaces or is ASCII support primarily intended to insure that dump terminals are controlled by WEX? If users are to work from an ASCII terminal, full support from clients is required.

Page 5-7

The document states that SVID UNIX will be used. How will compliance to SVID be verified?

The document makes the statement: "single software architecture will be used across multiple hardware platforms". What platforms (other than the MASSCOMP) will the software be executed and used on? What are the plans for demonstrating this capability?

The "tightly-coupled, multi-processor, shared memory" approach requires a specific, somewhat non-standard workstation configuration. The only other configuration which this design supports is use of workstations or X terminals as dedicated displays. However, even this configuration is in jeopardy, as TCP/IP (upon which X depends) is not available on the GP LAN. Also, has it been demonstrated that the shared memory approach is the only one which is feasible? Has it been proven that a distributed concept using the network would in fact be too slow?

Page 5-8

Using the "uwm" window manager is normally a good choice as it is robust and has little impact on the user environment. Unfortunately, uwm is markedly different from the window manager used by MOTIF (towards which WEX will migrate). Much of the MOTIF "look and feel" is due to its window manager, which places a number of objects (for iconify, resize, move, expose, etc) on every window present on the display. This is the primary means whereby a user manipulates windows, whereas uwm depends on pop-up menus and

user-defined "hot-keys" to provide similar functions. The point is that if the migration is towards MOTIF, it is advisable to select a window manager which eases the transition.

What does the statement "applications designed to run with or without the window manager" imply? It is assumed that this implies addition of functionality to allow windows to intelligently react to asynchronous window events, such as resizing, movement, and exposure. This functionality is expected and is not complete unless each window is able to "intelligently" deal with such events.

Although WEX does not "preclude the use of TAE", it is not understood why a tool like TAE is not used for development of the user interface. This would make prototyping much easier and allow rapid implementation of changes. Although it may prove difficult to learn TAE and port clients to it, this may save time in the long term and improve the quality of the WEX user interface.

Page 5-9

When running an X terminal (or a workstation as such), the server will be resident on the display processor. The statement "WEX is limited to managing X servers on resident workstation" indicates that this configuration would be impossible to use (assuming TCP/IP or some equivalent is available for X protocol communication).

The statement "Majority of WEX/WSA applications are designed/implemented to employ the X Window System" is confusing. Does this imply that some clients will not use an X Windows based interface and will use an ASCII based interface instead?

Page 5-13

What is the timeframe for migration to MOTIF? Will this be part of the 2.5 delivery or will it be included in 2.7? With as much attention as MOTIF receives in this document, it appears that it will be part of 2.5

Page 5-14

Use of the HP widgets is rationalized by referencing use by TAE, yet TAE is not used for WEX user interface development.

The statement "By ensuring that our clients do not have dependences on a particular Window Manager" is incorrect, as presentation of title bars depends on the window manager not presenting its own title bars.

The document states that users will be allowed to use their favorite window manager. This will make the design of WEX clients more difficult as they will have to determine whether or not it is necessary to present title bars.

For more information on these points, refer to the discussion under the heading of "Page 5-52".

Page 5-15

The diagram indicates that 250K of shared memory is required for "WEX buffers". Why is so much memory required? What is the breakdown of memory in this requirement. Is this memory statically allocated?

Page 5-22

Will all WEX clients (and applications) include processing to handle the SIGTERM signals sent during the recycle process? Typically an application includes signal processing code to trap and process such signals. If not present, the application will simply terminate without cleaning up.

Page 5-27

The document states that 60K of shared memory is allocated (in addition to the 250K) for each advisory terminal. Is this memory statically allocated or dynamically as required by the message load? Why is it necessary to save messages in memory? Once messages are displayed, couldn't they be saved in a file? This would save memory and still provide good response for occasional viewing of older messages.

Page 5-33

It appears that the EXitdaemon is required to clean up after users who do not terminate their attachment to WEX. A more effective approach may be to have these checks performed asynchronously (during a EXwexinit call) rather than cyclicly.

Page 5-35

The document states that "the mouse will not be required to login". Will all clients provide equivalent keyboard functionality so that users are not forced to use the mouse?

Page 5-38

Are the Tab/Return and Cnrl/Shift/Return function key mappings available for all WEX clients?

Page 5-43

Why is UNIX accounting initialized? UNIX user and process accounting adds overhead to normal system operation. This function should not be used unless there is a requirement for the accounting information.

Is the user notified of UNIX mail, the ISO X.400 mail, or both? Is the user even allowed to use regular UNIX mail due to the new mail system?

Page 5-45

Why are background processes killed on an "all or nothing basis"? Why is the user not allowed to kill some processes and leave others running?

Page 5-50

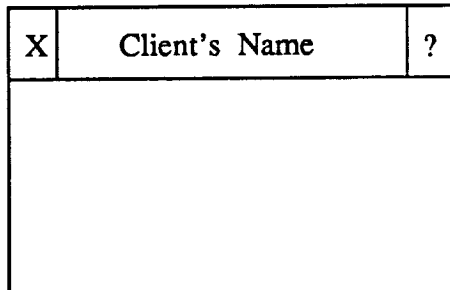
Note that by changing some defaults, users can dramatically change the operation of their own and WEX clients. For example, if password text is hidden by using the same foreground and background text color, the user can override this and thereby cause the text to be displayed.

Page 5-52

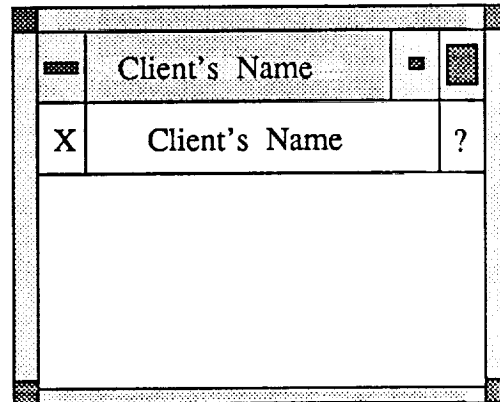
It is noticed that the WEX user interface includes a standard title bar. This is an excellent idea when using a non-obtrusive window manager such as uwm (or when not using a window manager at all). Unfortunately, this title bar will be redundant when other window managers are used, as they present their own title bars along with a subset of common functions. For example, consider the following example which shows a clients appearance with the uwm window manager and then with the MOTIF window manager:

Comments on WEX 2.5 Critical Design

Appearance (uwm)



Appearance (MOTIF)

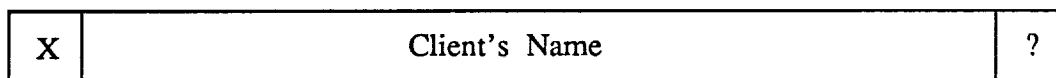


As this example shows, the title bar and some functions become redundant with the MOTIF (and most other common) window managers. A solution is to mandate that all users use one window manager and design the clients accordingly. In this case, you could force all users to use the uwm window manager and present a title bar in each client. Alternatively, you could force all users to use a different window manager (rtl, twm, awm, etc) and assume that the title bar and function responsibilities will be handled by the window manager.

If it is decided that users may use the window manager of choice, then WEX clients should sense the window manager running and present the appropriate interface (whether or not to present a title bar).

In this discussion of title bars, the important point is that all clients have a standard appearance and that common functions are always located in a consistent location. More important than the title bar is the location of common client-specific functions such as terminate and help. It is not necessary for clients to handle operations such as iconify, as this will be handled by the window manager. Rather, a client should present standard functions which are client-specific. It is especially important for an client to present its own terminate function, as using the standard window manager function will often send an unexpected "terminate" or "kill" signal which is not adequately handled.

If all WEX clients are to present a title bar, it is suggested that it appear as shown below:



Where: "X" - Terminate client function
"?" - Help function

In the instances where a title is not required, the terminate and help functions should be presented in a standard location.

Does "application busy clock" indicate that a client is active or process-bound (not available due to processing)? A more common means of achieving this is to change the mouse cursor to an image indicating that the application is busy (hourglass, watch, etc.).

It is not clear what scheme will be used to display local messages. Will each client have its own message area (not recommended) or will pop-up windows be extensively used? For clients with a large amount of output (especially multi-line output), special-purpose output windows are advised. For normal output, pop-up windows are preferred (note that pop-ups must not halt operation of critical applications).

All WEX clients should adopt a standard mechanism for presenting user commands. This could be via a command line at the top of the client from which pull-down menus are accessed. Another option is to place all available commands as buttons along the top or side of the screen. The decision should be based on what users prefer, what is handled well by the HP widgets, and what interface is used by MOTIF.

Page 5-54

The document indicates that several applications do not require changes (no port to X Windows). Is this because the applications have no user interface? If they do have an interface (and are not temporary applications) then they should be ported to X Windows.

Page 5-56

How are the "shift change" and "update rate" pop-up windows selected from the information client?

Page 5-57

If iconified and an advisory is received, will the advisory client become active and display the message? This feature should be available and it should be possible to enable and disable it.

Page 5-61

The WEX Host/Flight list should be a function of the information client. The information client should provide this information as a pop-up window.

Page 5-64

What is the raw format of the help text? Will it be the same as that used for UNIX "man" (nroff with -man macro support) or in straight text format? If in man format, the text could be viewed via the command line. Note that help on clients should be available via the command line. This allows users to access help via familiar utilities.

Is the help text context-sensitive or will the user simply get a large amount of text which must be scanned?

Page 5-71

It appears from this discussion that WEX services are required if the user needs "LAN access utilities". Does this include real-time data acquisition? Some users may require data acquisition but not the full set of WEX services.

3.0 CM Comments

Page 6-11

The document states that files which cannot be linked across file systems will be copied. Files across file systems can always be linked with "symbolic links" (also called soft links).

Page 6-13

The document states that the "chmod" command is front-ended to prevent making a file executable. What protection is used to prevent the programmatic call from performing the same function?

Page 6-29

The user interface for the EXcmmenu client wastes a great deal of screen space. Also, the commands appear to be placed in an awkward location.

Pages 6-31 - 6-36

These pages present several of the windows used by the EXcmmenu client. Note that the user interface for each of the CM sub-functions uses a different approach for presentation of commands.

Page 6-43

The document describes the "replace" option for download via a CDL. Does this option apply to the entire download as a whole or can it be responded to for each individual file?

Page 6-63

Why is no X Windows interface provided for the EXcm_copy and EXcm_erase functions. Also, will any command line interfaces be provided for the functions provided by EXcm-menu?

Page 6-68

The workstation initialization status display is a good idea, but why is the information presented in a special client. At this point, the xterm window (presented at login) should be available for display of these messages.

Appendix - C
Report on
Graphics Use in MCCU

Abstract

In order to satisfy the diverse graphics requirements of the MCCU, it will be necessary to utilize several different graphics software tools. The key will be to use tools which are portable, compatible with X Windows, and best suited to the requirements of the associated application. From a high-level viewpoint, this will include a User Interface Language (UIL), an interactive display builder, and a graphic plotting/modeling system such as GKS or PHIGS.

1.0 Introduction

The graphics requirements of the Mission Control Center Upgrade (MCCU) environment are quite diverse. The requirements range from simple text display to user interaction to complex 2 and 3-dimensional modeling. These requirements are most effectively satisfied with a combination of graphics software tools, each utilized for those applications in which it is most appropriate.

Satisfying a diverse set of requirements and at the same time using standard graphics software, has been a difficult task in the past. If portability was required, requirements were often met by utilizing graphics software tools which were not ideally suited for the application. For example, it is not appropriate to use a high-level graphics system such as GKS or PHIGS to display text or present user interface objects (such as colored rectangles which change orientation when selected).

In the near future, all levels of standard graphics software will be available. This software is already available, but in many cases is public domain and is unsupported. When all software is available as supported products, it will be possible to meet all graphics requirements of the MCCU with standard tools which are efficient for development and execution.

2.0 X Windows and Graphics

The foundation of all graphics generated in the MCCU will be the X Windows system. In the near future, all graphics requests will ultimately go through the X Windows system, thus allowing all graphic input and output to be transparently shared across the network. The X Windows system consists of a client-server model in which every display is managed by an X Windows server. A client communicates display requests to the server (which may or may not be on the same system) via a standard protocol (the X protocol). Every graphics request, no matter at what level it was initiated, will ultimately be translated into the appropriate X protocols and routed to a server for display. This routing from client to server may be local or across the network, thus allowing graphics to be generated on a host system (possibly a data or compute server) and then displayed on any workstation in the network.

One problem with the X Windows system is that benefits of the standard protocol are only realized when using graphics over a network. For local graphics interaction, the translation and routing of the X protocol reduces the performance of graphics. Many vendors are addressing this problem by using shared memory or some other fast medium for routing of client-server protocol. This increases performance, as a medium such as shared memory is significantly faster than typical network communication methods (TCP, ISO). However, overhead still remains due to the protocol translations.

A more serious problem is that the X protocol currently supports only 2-dimensional graphic functionality. Using 3-dimensional functions is difficult as they do not closely trans-

late to available X Windows protocols. To solve this problem, vendors of 3-dimensional software (such as PHIGS) will cooperate with the X Windows system (work within its windows), but actually bypass the client-server routing and directly access the native system graphics. This allows local use of such graphics software, but precludes its use over the network. Fortunately, a solution to this problem is in the works in the form of standard extensions to the X protocol. These extensions are called PEX (PHIGS Extensions to X). They implement 3-D functionality in the X protocol, thus allowing 3-D graphics requests to be efficiently processed.

3.0 Contents of the X Windows System

X Windows is a relatively new standard and as such, does not provide a complete set of the desired (or required) graphics development tools. As defined by the X Consortium, an X Windows system as provided by a vendor must provide the following applications and libraries:

- X Windows server,
- XLib library, and
- Xt library.

This combination of applications and libraries is not adequate to efficiently support all graphics requirements. The Xlib function calls operate on a very low level and are quite difficult to use. The Xt function calls define a standard whereby user interface objects (called "widgets") may be defined and manipulated. It does not include any widgets or a higher level library for presentation of user interfaces.

Note that another requirement for effective use of the X Windows system is a small set of clients, including a terminal emulator, window manager, environment manipulation applications, and other applications. Such clients are normally provided with an X Windows system, but are not part of the standard as defined by the X Consortium.

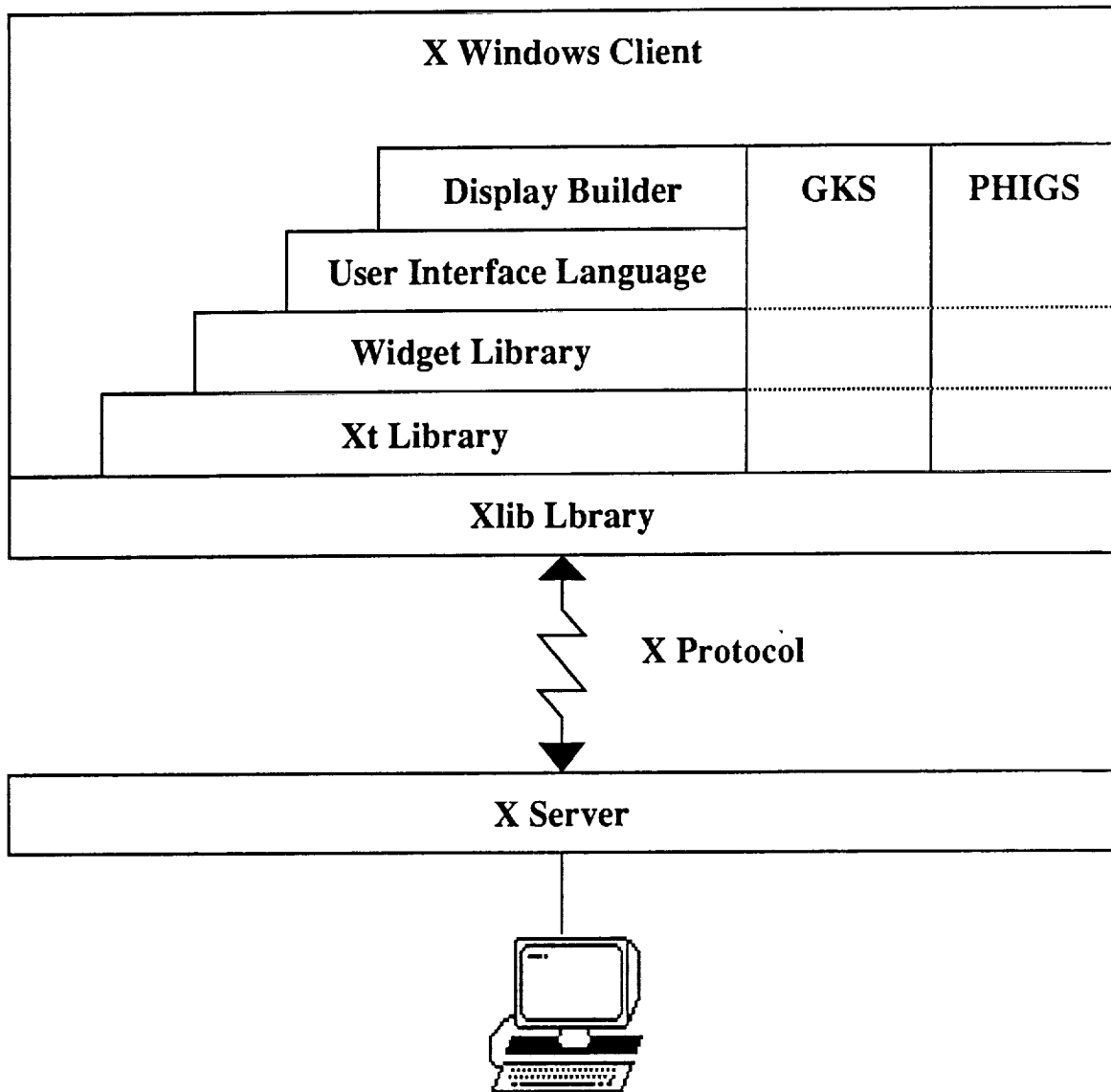
To provide a complete set of graphics development tools, additional libraries and applications must be layered upon the Xlib and Xt libraries. These include the following:

- A library of widgets,
- a User Interface Language (UIL),
- an interactive display builder, and
- a high-level 2 or 3-dimensional plotting/modeling system.

The entire X Windows system including the server, basic libraries, and high-level tools is graphically summarized in Figure 1.

Most users in the MCCU environment are familiar with the term widgets. Widgets are individual user interface objects such as command buttons, menus, sliders, etc. Using the Xt library and a set of widgets greatly simplifies presentation of user interfaces. Widgets operate at a high-level and automatically handle typical window system requirements such as cursor redefinition, resizing, and highlighting of active objects.

Although widgets may be directly used by the programmer, it will be more efficient to utilize a UIL to define a client's display interface. A UIL allows a programmer to completely



Note: GKS and PHIGS provide their own user interface objects for the various logical input devices supported. Ideally, the GKS and PHIGS implementations will utilize widgets and the Xt library to support such functions. This of course will be implementation-dependent.

Figure 1 - X Windows Software Structure

define a user interface without actually developing or modifying code. This provides the following advantages:

- Significantly reduces development time,
- allows user interfaces to be rapidly prototyped,
- allows simple and global updates to be rapidly implemented, and
- separates user interface definition from actual code (important for CM).

Note that the final advantage is important as it will allow changes in the user interface to be implemented without affecting the actual code. The UIL description would still be certified, but this would be simpler than recertifying an entire application.

Although a programmer can design a user interface by "coding" in the UIL, it is more efficient to utilize a tool which allows the interface to be interactively designed. This display builder will allow the programmer to interactively select and place user interface objects (widgets) on a blank form. The programmer is thus able to design and arrange the interface and receive immediate visual feedback. The end result of this process is a file containing the corresponding statements in the user interface language.

The final applicable graphics tools include 2 and 3-dimensional plotting and modeling systems. This includes GKS (Graphic Kernel System) and PHIGS (Programmers Hierarchal Interactive Graphics System). From a superficial standpoint, the functionality provided by these systems is the same as the low-level X Windows libraries. However, upon closer examination, these systems have a great deal more functionality. The primary advantage of these systems is that they allow objects (whether simple plots or complex models) to be defined and then scaled, rotated, and transformed in other manners. For example, a 2-D model could be scaled to simulate it moving closer to or farther from the viewer. A 3-D model could be rotated to view other sides or to view affects of light on its different surfaces. Whereas in X Windows, integer device coordinates are used, GKS and PHIGS allow floating point coordinates, the ranges of which are defined by the programmer. The transformations from these coordinates to the actual device coordinates require a large number of floating point calculations. This amount of floating point processing makes the overhead of using GKS or PHIGS too great for simple drawing applications, such as displaying text, drawing lines, or presenting a user interface.

4.0 Meeting MCCU Graphics Requirements

Although the graphics requirements of the MCCU are quite diverse, they may be summarized into a few categories. These include the following:

- Window-based user interface,
- high-performance, high-flexibility data-driven display, and
- plotting and modeling.

The key to meeting MCCU graphics requirements is to use the appropriate graphics software for each application. Providing a window-based user interface is achieved by using the Xt library and a set of widgets. Layered on top of this will be the UIL and the interactive display builder which utilizes it. These tools will be used by all applications which present a graphic user interface. This includes system applications (such as WEX) as well as user applications.

The display builder application (and underlying UIL) must be flexible enough to support all MCCU requirements. This not only includes development of displays and placement of user interface objects, but also a way for users to associate their data with output objects (widgets). For example, the user may require some subset of data (real-time or other) to be displayed at a defined interval. The output may consist of text and/ or objects which graphically display results (such as a dial, a gauge, etc). The widgets provided for this purpose must balance performance with visual and graphic flexibility. For example, a widget must be provided to display and update a large number of data values (>100) in a manner which does not unduly affect the performance of the system. Other widgets will be provided for less dynamic or voluminous data, in which more graphic presentation is desired (it is easier to recognize a critical condition by examining a graphic dial which is approaching a marked level).

The combination of user interface and data demands make it difficult for any existing graphics tool to satisfy the requirements. What is required is a general-purpose UIL and display builder which allows development of client displays containing the required set of user interface widgets, a number of which may be automatically driven by external data. Some of the existing or planned systems which support a subset of this functionality include the following:

- Display Builder application developed by Ford Aerospace,
- the Data-Views drawing system,
- the Open Software Foundations (OSF) MOTIF system, and
- the Transportable Applications Environment (TAE)

Each of these systems has its advantages and disadvantages. SwRI's recommendation is to use one of the systems (or an equivalent) and either modify it to provide all required functionality or integrate it with other products to achieve the same goal. The end product of such a system should be a language which is used by actual code to present the user interface and drive the display of data.

The UIL and display builder will not satisfy all MCCU requirements. There will remain applications which must construct complicated graphic plots and/or models. For such applications, if the requirements are too complex for one of the X Windows software tools, a system such as GKS or PHIGS should be used.

There are two basic approaches for integrating GKS or PHIGS graphics into an X Windows application. The first is for the application to exclusively use GKS or PHIGS and use their inherent capabilities for user interface. As described in Figure 1, GKS and PHIGS provide their own user interface functionality. However, using this approach will most likely (depending on the implementation) make the applications interface different from others in the environment. The second approach would be to use the display builder to design a screen which includes a blank "canvas" area, which may be as large or small as is desired by the programmer. This area will be used as a virtual workstation into which the GKS or PHIGS output may be directed. In this approach, X Windows functions will be used for user interface requirements. The second approach is preferable, but requires additional functionality in the UIL and display builder.

In SwRI's understanding, the requirements for GKS or PHIGS are not common in the MCCU environment. This however may change in the future and it does not make sense to

design other software which will preclude the use of these graphics tools. If the other graphics tools meet all requirements, only a small subset of users will actually require use of GKS or PHIGS. Such instances should always be reviewed to determine if the software is really necessary.

Note that there remains a decision as to whether GKS or PHIGS will be used for the related requirements. Although the packages are conceptually similar, there are some major differences. GKS is currently a more stable graphics standard. It is supported by more vendors and is significantly faster than PHIGS. It provides adequate 2-dimensional capability for static display of graphics. On the other hand, PHIGS is a relatively new standard and will by its very nature, be slower than GKS. However, PHIGS provides 3-dimensional capability and more importantly, allows dynamic changes to images. When using GKS or PHIGS, a programmer will normally create a segment (GKS) or structure (PHIGS) which defines some graphics object. In GKS, if that segment is to be changed, it must be destroyed and then completely recreated with the required changes. In PHIGS, it is possible to specify only the changes, thus making it more efficient for very dynamic images.

A final advantage of PHIGS is that its described functionality will be added as an extension to the X Windows protocol. PEX (PHIGS Extensions to X) will allow PHIGS graphics requests to be directly implemented in the X protocol, thus improving performance and insuring that the graphics will work over a network. In the case of GKS, the requests must be converted to the appropriate X Windows protocol. Essentially, GKS is a good short-term solution for applications which will not require 3-D or dynamic image functionality. PHIGS however, for the reasons outlined above, is a better long-term solution.

Appendix - D
Work Sheets for
POSIX 1003.4 Real-time Extensions

OSID WORKSHEET

Control Number

Title

Optional features in POSIX 1003.4.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) Global
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

All of the major functions of POSIX 1003.4 are termed "optional". By definition, a strictly complying POSIX 1003.4 implementation would only be required to provide the functional interfaces (as opposed to any actual functionality).

Resolution

Add a new implementation definition which identifies a POSIX 1003.4 implementation which provides all listed functionality.

Rationale

Adding a new definition would reduce confusion for individuals trying to procure a system on the basis of POSIX 1003.4 compliance. The current definition may mislead individuals into procuring systems which do not adequately meet all requirements.

OSID WORKSHEET

Control Number

Title

Performance metrics.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) Global
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

The performance metrics need to be further defined for many of the 1003.4 functional areas. The definition should specifically state the metrics to be used and should provide algorithms for obtaining metric measurements.

Resolution

Complete the performance metric sections and introduce algorithms which explicitly show how the metrics may be measured.

Rationale

Most individuals involved in system procurement will be required to execute their own metrics on a system (as opposed to using measurements provided by the vendor). In the absence of a standard test suite, users will develop programs which measure the metrics. The amount of variability in these programs will be unacceptable unless the performance metrics are well defined and include basic algorithms. The advantage to algorithms (over code examples) is that they are language-independent. Actual code samples would be useful, but would be tied to a language (C or Ada).

OSID WORKSHEET

Control Number

Title

Pointers to static buffers.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 2.2.5
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

No POSIX 1003.4 function (or any POSIX function) should return a pointer to a statically allocated internal buffer.

Resolution

Add a paragraph which states that pointers updated and returned by POSIX functions will address dynamically allocated memory which may be directly used.

Rationale

A common problem in UNIX systems is that C function calls return pointers to memory statically allocated within the function. In order to safely use the data, the programmer must copy it to a new location before a subsequent call to the function. Designing functions which return pointers to dynamically allocated data allows more direct use (at the expense of forcing the programmer to free the memory).

OSID WORKSHEET

Control Number

Title

Changes to "General Terms".

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 2.3
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

There are several terms which must be modified or added in the "General Terms" section.

Resolution

Change or add the following terms:

- "binary semaphore" - (change) Change binary semaphore definition to describe the difference between binary and other (such as "counting") semaphores.
- "persistent" - (add) Persistence in this context refers to a special file (shared memory, semaphore) which remains available after the last close.

Rationale

These terms are required to improve understanding.

OSID WORKSHEET

Control Number

Title

Section 2 is not finished.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 2.2.4, 2.8, 2.9, 2.11
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

Many of the subsections in section 2 are incomplete. These sections are either blank or contain statements such as "The quick brown fox jumped over the lazy dogs".

Resolution

Complete the sections in question.

Rationale

These sections contain important information which must be completed for the final draft.

OSID WORKSHEET

Control Number

Title

Use of [ENOTSUP] and [ENOSYS] *errno* values.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 2.11.5
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

The description of the [ENOTSUP] *errno* value includes a description of the [ENOSYS] value. The description of [ENOSYS] is misplaced and is not correct.

Resolution

Describe the [ENOSYS] *errno* value in a separate bullet. Also, change the phrase "the implementation does not support all required functionality" to "the implementation does not support any required functionality".

Rationale

The description of the [ENOSYS] *errno* value is referenced inside the bullet for the [ENOTSUP] *errno* value. The description should be placed within its own bullet. In addition, the use of the word "all" is incorrect, as this error value indicates that the implementation does not provide "any" of the required functionality. This is opposed to the [ENOTSUP] *errno* value which indicates that only a particular function is not implemented.

OSID WORKSHEET

Control Number

Title

Binary semaphore persistence over a reboot.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☒ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 3.4.1.2, 5.4.1.2, 9.3.2.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The persistence of a binary semaphore over a reboot is implementation defined. This could result in non-portable code.

Resolution

Define persistence over a reboot in a standard manner. At worst, assume that the binary semaphore value is undefined.

Rationale

There is no benefit to allowing this behavior to be implementation defined. To be portable, an application must assume that a binary semaphore value is unusable after a reboot (worst case). To assume anything else would cause portability problems.

This comment applies to other IPC services which are implemented as special files and allow persistence (message queues and shared memory).

OSID WORKSHEET

Control Number

Title

Use of *read()*, *write()*, and *lseek()* on binary semaphores.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☒ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 3.4
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

Use of the *read()*, *write()*, and *lseek()* functions for a binary semaphore do not make sense.

Resolution

Describe such operations as undefined.

Rationale

Use of any of these functions does not make sense in the context for binary semaphores. The *read()* and *write()* function do not adequately relate to the wait (lock) and post (unlock) operations. The *lseek()* function has no use whatsoever.

OSID WORKSHEET

Control Number

Title

Inconsistent use of ANSI C.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 3.4.4.1, 10.3.1.1
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The function prototype for *semifwait()* does not use ANSI C format. This is not consistent with the majority of the document. A similar problem is found for the *fentl()* function in section 10.3.1.1.

Resolution

Update the function prototype to use ANSI C format.

Rationale

ANSI C allows a function prototype to directly include the types of all parameters (as opposed to being typed on the following lines). This format is useful and is used throughout the document.

OSID WORKSHEET

Control Number

Title

Use of the *unlink()* function on a binary semaphore special file.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☒ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 3.4, 5.4, 9.3
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The effect of the *unlink()* function on a binary semaphore special file is not described in the document. What would occur if another process (or a command) was used to remove a binary semaphore special which was locked or unlocked? Would a process which was waiting on a binary semaphore receive the appropriate notification (via *errno*) or would it suspend indefinitely?

Resolution

Define the effects of the *unlink()* function for the described instances.

Rationale

The affect of the *unlink()* function on persistent binary semaphores must be known to write portable code.

Similar problems apply to other IPC services which are implemented as special files and allow persistence (message queues and shared memory).

OSID WORKSHEET

Control Number

Title

Indication of unlock (post) of binary semaphore which is already unlocked (posted).

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 2.2.5
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The *sempost()* function does not indicate in any way in which the programmer can determine if the binary semaphore unlocked (posted) was already unlocked.

Resolution

Update the *sempost()* function to return an indication that the binary semaphore was already unlocked. This return should be transparent such that programs not interested in this case would not be affected.

Rationale

This ability is important in the 1003.4 implementation, as the "holder" of the binary semaphore (the locking process), will not be the only process which is allowed to unlocked. The described indication would allow a process to determine if another process inadvertently unlocked the binary semaphore, thus averting a potential deadlock or race condition.

OSID WORKSHEET

Control Number

Title

Typo in process memory locking section.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 4.3
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input checked="" type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The process memory locking section, line 35 contains a typo. "Terminate a Process" is written twice.

Resolution

Correct the document.

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

Typo in shared memory section.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 5.3
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The shared memory section, line 28 contains a typo. The phrase "defined. binary" should be changed to "defined, binary".

Resolution

Correct the document.

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

Use of *read()*, *write()*, and *lseek()* functions for shared memory special files.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☒ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 5.4
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The behavior of the *read()*, *write()*, and *lseek()* functions should be defined. The document describes their behavior as implementation-defined.

Resolution

Define the behavior of these functions.

Rationale

These functions are reasonable for shared memory files and their behavior should be defined. Although shared memory will most commonly be mapped and accessed directly, it may be more convenient for certain programmers to use familiar *read()*, *write()*, and *lseek()* functions. If it is determined that this is unreasonable for general implementation, then the behavior should be undefined. At any rate, it is not reasonable to leave the behavior as implementation-defined.

OSID WORKSHEET

Control Number

Title

Implementation specific priority scheduling algorithm.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 6.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|---|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input checked="" type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

Three scheduling types are required, but only two are defined. The third designated by SCHED_OTHER must be implemented to conform, but the algorithm used is implementation specific.

Resolution

Specify in OSID that SCHED_OTHER shall implement an "aging" scheduling algorithm.

Rationale

This implements the intent of 1003.4 which is to allow a separate non-real-time scheduling algorithm.

OSID WORKSHEET

Control Number

Title

Negative priorities may cause problems.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☒ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 6.3.1.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|---|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input checked="" type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

Setpriority() and *getpriority()* returns the priority as an *int*. This is a problem as certain implementations may allow negative priorities.

Resolution

Specify that priorities of -1 are not allowed.

Rationale

Negative priority levels are allowed as an implementation-defined feature. This is a problem as certain priority functions return -1 to indicate an error condition, making it impossible to differentiate a -1 error from a -1 priority.

OSID WORKSHEET

Control Number

Title

Function to return current event class mask.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 7.4.2.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|---|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input checked="" type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

There is no straight-forward manner to retrieve the current event class mask. The *evtprocmask()* function provides this value, but via an odd combination of parameters.

Resolution

Add a new function which returns the current event class mask

Rationale

The manner in which the event class mask is retrieved via the *evtprocmask()* is a "back door" approach. A more simple and straight-forward function is required. A simple macro designed around the *evtprocmask()* would be suitable.

OSID WORKSHEET

Control Number

Title

Function types for *evtsetjmp()* and *evtlongjmp()*.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 7.4.6.1
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|---|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input checked="" type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The types shown for the *evtsetjmp()* and *evtlongjmp()* functions are incorrectly defined as *int*.

Resolution

The return values for the *evtsetjmp()* and *evtlongjmp()* functions should be changed to *void*.

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

System timer setting.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☒ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 8.3.1.1
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input checked="" type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The privilege to set a system timer is implementation-defined. Non-superuser's may or may not be allowed to set a system timer, as defined by the implementation. This leads to non-portable applications.

Resolution

Define the privilege required to set a system timer.

Rationale

A system timer is not normally available for user modification on multi-user systems. The most reasonable behavior is to limit system timer modification to the superuser. Alternatively, non-superusers may be allowed to set the timer (which would not make sense on a multi-user system). At any rate, the behavior needs to be consistent in order to allow development of portable code. The current behavior would force an application to assume that superuser status is required to set the timer.

OSID WORKSHEET

Control Number

Title

Timer section typos.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 8.3.1.2, 8.3.2.2, 8.3.4.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input checked="" type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The listed section includes the type "*struct*timercb" which should be "*struct* itimercb".

Resolution

Correct the document. Note that section 8.3.4.2 includes two instances of the typo.

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

Timer resolution.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 8.3.1.1
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input checked="" type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The document does not describe how a timer resolution is expressed. It is assumed that the resolution will be expressed in some increment of nanoseconds (1, 1000, 1000000, etc); however, this is not clearly stated.

Resolution

Clearly state the manner in which the timer resolution is retrieved.

Rationale

The resolution provided by the current implementation's timers is critical to all timer functions. The manner in which this information is retrieved must be clearly stated to allow other functions to be understood and effectively used.

OSID WORKSHEET

Control Number

Title

itimercbp / *itimercb* parameter typo.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 8.3.2.1 and 8.3.2.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input checked="" type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The "Synopsis" section lists a structure instance named *itimercbp*; the discussion in the following "Description" section uses a structure instance named *itimercb*.

Resolution

Update the *itimercbp* value in the function prototype, as it is this value which appears to be incorrect.

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

Use of timer values which are not multiples of resolution.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 8.3.4.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input checked="" type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The document does not describe what is done with timer increments which are not even multiples of the timer resolution.

Resolution

The exact treatment of such values should be defined. This may be via rounding up, truncation, or error return.

Rationale

The most appropriate behavior is difficult to select. Rounding up or truncation may cause problems for applications requiring critical timing. Returning an error is probably the best solution, but would require more work on the part of the programmer. The programmer would be required to retrieve the system resolution and only use valid multiples of this value.

OSID WORKSHEET

Control Number

Title

nanosleep() function.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 8.3.5.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input checked="" type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

In many (most) implementations of the *nanosleep()* function, the timer resolution will not be able to sleep for small numbers of nanoseconds or accurately for odd multiples of the system timer resolution.

Resolution

Behavior of the *nanosleep()* function should be defined for such instances.

Rationale

The behavior should specify rounding up, truncation, or error return as consistent with the behavior of all system timers.

OSID WORKSHEET

Control Number

Title

Adding a function to purge all messages from a message queue.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 9.
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

One of the open issues is to add a "purge all messages from the queue". This would be a useful function.

Resolution

Define a function which allows a process to selectively purge messages in a queue.

Rationale

The direction of this open item is to add a function which returns the number of messages in the queue. Although useful, the programmer will still be required to purge the applicable messages from the queue. It would be more convenient to provide a set of functions which selectively purges messages based on *pid*, *type*, and other values. This function could purge all messages within a category (*pid*, *type*, etc) if the corresponding value is a pre-defined constant (not 0 as this would be dangerous).

OSID WORKSHEET

Control Number

Title

Name of the *mq_errno* member.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 9.3.1
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The name of the *mq_errno* member of the message control block structure *mqcb* should be renamed to reflect its relationship to asynchronous event errors.

Resolution

Rename the *mq_errno* member to for example, *mq_aserrno*.

Rationale

The existing *mq_errno* member name does not reflect that it pertains only to errors from asynchronous reads.

OSID WORKSHEET

Control Number

Title

Operation of *mqsend()* function with *MQ_ASYNC* flag.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 9.3.6.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input checked="" type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

Using the *MQ_ASYNC* flag in an *mqsend()* function states that an asynchronous event notification will occur when the message has been received. It is assumed that event generation is automatic; however, this is not clearly stated in the document.

Resolution

Define the behavior more clearly.

Rationale

The behavior must be clearly stated in order to develop portable code. If the programmer assumes that the system will generate the message and it never occurs, then the message sender will never receive notification. On the other hand, if both the system and the receiver generate events, the sender will receive duplicate notifications.

OSID WORKSHEET

Control Number

Title

Performance metrics for Synchronized I/O.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 10.5
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input checked="" type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The performance metrics Synchronized Input Time and Synchronized Output Time require the transfer of 1 megabyte of data. This creates a problem of manufacturers using very large caches to fool the metric. A better approach would be to specify a data size at least N*the largest cache in the system or specify that all caches and buffers shall be cleared before the performance metrics are executed.

Resolution

Add a statement to the OSID that all caches and buffers shall be cleared before performance metrics are executed.

Rationale

This will ensure performance metrics will be comparable between vendors.

OSID WORKSHEET

Control Number

Title

Clarification in *acancel()*.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 11.4.5.3
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input checked="" type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

The meaning of a return value of 0 for *acancel()* is confusing. Line 259 states that a return value of 0 means the requested operations were canceled. This implies the event has occurred before the return from the function. Line 263 states that event notification is not given for asynchronous I/O cancellation.

Resolution

Rewrite line 263 to read "Returning from *acancel()* serves as event notification."

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

Type in introduction to Real-time Files.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 12.1
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input checked="" type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

A typo appears on line 32 of paragraph 12.1. The line should read "with no changes necessary", not "with not changes necessary".

Resolution

Correct the document.

Rationale

N/A.

OSID WORKSHEET

Control Number

Title

Comments on Performance metrics in Real-Time files.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☒ OK
- ☐ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) 12.5
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|---|--|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input checked="" type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input type="checkbox"/> 12. Other |

Description

In the Real-Time files section 12.5, the performance metrics Maximum Transfer Rate for Read Operations and Maximum Transfer Rate for Write Operations specifies the number of data bytes transferred by a percent of the available file system. This makes no mention of cache sizes and can give misleading results.

Resolution

Add a statement that all caches and buffers shall be cleared before performance metrics are executed.

Rationale

This will ensure performance metrics will be comparable between vendors.

OSID WORKSHEET

Control Number

Title

Structures described to be implementation defined.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☐ Modification of Strictly Conforming POSIX
- ☒ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) B.1.2.2
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

Paragraph B.1.2.1 describes several structures as implementation-defined. This means that each implementation can define different structures required by threads. For portable applications, this is unacceptable.

Resolution

Add a statement which specifies that all structures in B.1.2.1 shall only be used in the default state.

Rationale

This will allow portability of code.

OSID WORKSHEET

Control Number

Title

General comments on threads.

Applicable Profile(s)

- ☐ Real-Time (Onboard-DMS)
- ☒ Real-Time (Ground)
- ☐ Other

Classification

- ☐ OK
- ☒ Modification of Strictly Conforming POSIX
- ☐ POSIX Extension Required
- ☐ Non-POSIX Extension
- ☐ Defn. of Implementation-Defined Behavior

POSIX References

- ☐ 1003.1 Paragraphs(s)
- ☒ 1003.4 Paragraphs(s) B
- ☐ 1003.6 Paragraphs(s)
- ☐ 1003.8 Paragraphs(s)
- ☐ Other Paragraphs

Subject(s)

- | | | |
|--|--|---|
| <input type="checkbox"/> 1. Process Env. | <input type="checkbox"/> 5. Timers | <input type="checkbox"/> 9. Networking |
| <input type="checkbox"/> 2. Process | <input type="checkbox"/> 6. Files/Dir. | <input type="checkbox"/> 10. System Adm. |
| <input type="checkbox"/> 3. Scheduling | <input type="checkbox"/> 7. Input/Output | <input type="checkbox"/> 11. Security |
| <input type="checkbox"/> 4. Events | <input type="checkbox"/> 8. IPC | <input checked="" type="checkbox"/> 12. Other |

Description

The entire section on threads is still in flux. There are several open issues which will take time to resolve. Threads should be reviewed again at a later time.

Resolution

Track threads in the coming 1003.4 drafts.

Rationale

Threads are an important feature of 1003.4 and should be tracked to ensure their usability for ground support.

Appendix - E
Discussion for
AIS Requirements in
POSIX 1003.6 Security Extensions

1.0 Introduction

This document interprets the requirements in Johnson Space Center's Automated Information Systems Security Plan (AIS) and determines if the requirements are met by the POSIX 1003.6 draft 3 extension for standard security (henceforth termed "POSIX 1003.6"). During the review of the AIS document, most of the detailed computer system-related requirements were found in Chapters 11 and 13. Some of the introductory chapters include high-level requirements which were later expanded in Chapter 11. For this reason, chapters 11 and 13 were exclusively used as the basis for the comments in this document. The basic organization of this document is to interpret each computer system-related requirement in the AIS and then describe whether or not the requirement is satisfied by POSIX 1003.6.

2.0 General Comments

From the review of the AIS, it appears that JSC will require the Discretionary Access Controls (DAC) and auditing. It does not appear that Mandatory Access Control (MAC) is required, as this is primarily required for systems which support different sensitivity levels of data. The AIS did not clearly indicate that this was necessary.

POSIX 1003.6 is not complete and does not yet address a number of important functional areas. While the Discretionary Access Controls (DAC) and auditing are fairly well developed, the Mandatory Access Controls (MAC) are incomplete and the Privileges section is effectively blank.

The auditing section of POSIX 1003.6 is fairly well-defined but has a limited scope. It does not address the manner in which auditing is enabled for users, files, or applications. It is not clear whether such functions should be part of POSIX 1003.6 or are more appropriately placed in the POSIX 1003.7 (System Administration) specification. POSIX 1003.6 depends a great deal on the POSIX 1003.7 (System Administration) specification. It is not clear whether administration of the security system will be part of POSIX 1003.6 or POSIX 1003.7.

POSIX 1003.6 does not address the role of the root (or superuser) user. The interaction between this privileged ID and the security system is undefined. It is not clear if there are multiple privilege levels (hierarchical system) or simply a single privileged ID which has access to the entire system (including security).

POSIX 1003.6 has significant effects on the POSIX 1003.4 (real-time) specification. In addition to the basic conflict between security and real-time response, it is unclear of the affects of DAC and MAC on the features provided by POSIX 1003.4, including message queues, shared memory, semaphores, timers, events, etc.

3.0 Detailed Comments

This chapter lists the section number and label for each of the relevant discussions in Chapters 11 and 13 of the AIS, provides an interpretation, and indicates whether or not the requirement is met by POSIX 1003.6.

11 Security Standards

11.3 Operating Systems

11.3.1 Operating System Controls

Interpretation

This capability will be provided via basic owner, group, and other permissions for read, write, and execute. Permissions will be attached to all objects (files and applications) for which protection is required. It will be the responsibility of users and administrators to assign and maintain the appropriate access controls.

Requirement

POSIX 1003.1 provides the owner, group, and other permissions for read, write, and execute. POSIX 1003.6 provides additional access control via Access Control Lists (ACLs), which are part of Discretionary Access Control (DAC). ACLs allow objects to include lists of user, group, and other permissions.

11.3.2 Clear System Programs

Interpretation

The AIS does not indicate when the "clear" operation must take place. It is assumed that this clear function will take place when the object is moved or removed from its current physical location. This requirement is applicable to all forms of random access storage, including disks (for when an object is removed/moved) and primary memory (when an application explicitly or implicitly [via termination] frees memory space).

This is a reasonable concern, as in most implementations, it is possible to allocate disk or memory space without actually clearing or writing any data. In this instance, the old sensitive data could appear in another object (a file or memory buffer). An application could repeatedly allocate disk or memory space in an attempt to locate sensitive data.

This requirement will most appropriately be provided via a remove function which will clear (zero out) space occupied by sensitive data. Note that it requires an order of magnitude more time to clear data space.

Requirement

POSIX 1003.6 references the term "object reuse", which indicates that objects (files, disk space, etc.), are reusable if they do not include any residual data. The actual mechanism for this was not found in the specification. This should be a feature of MAC, which is used on systems with multiple sensitivity levels of data.

11.3.3 Shutdown and Restart

Interpretation

It is not clear what this requirement entails. Systems which have failed or are down for normal maintenance tend to be very vulnerable (at least UNIX systems). Unless special steps are taken, it is often possible to boot from an alternate location (such as a tape, secondary disk, or floppy). This is an obvious problem as the software on the alternate media would not be constrained by the security system. Another security problem is that many

UNIX systems allow single-user mode to be initiated (booted) without a specification of a password.

Requirement

POSIX 1003.6 does not specify the security provided outside of the normal POSIX operating environment. Security outside of the operating system is inherently implementation dependent and would be difficult to specify the interfaces in a standard. This is however a valid requirement and should be addressed in either POSIX 1003.6 or 1003.7.

11.3.4 Recovery Management

Interpretation

Functions which need to be recovered after a system has failed should be defined when the system is designed so that any redundant information which is necessary for recovery and historical log keeping facilities can be built into the system.

Requirement

POSIX 1003.6 does not provide an interface for recovery management. This subject may be more appropriately covered in System Administration, 1003.7.

11.4 JSC Standard On Controlled Access Protection

Interpretation

Systems containing sensitivity level 2 or 3 objects will provide individual authentication and accountability by use of unique user IDs and passwords. The system will prevent unauthorized access or modification of this information.

Audit trails will be generated for all activity on privileged IDs and for user-selectable periods as required for suspected violations and spot checks. It must be possible to enable auditing for a given user or object.

User access to objects will be provided by permissions and ACLs. Maintenance of permissions is the responsibility of the user and system administrator.

Requirement

POSIX 1003.6 includes the concept of sensitivity levels via the Mandatory Access Controls (MAC). MAC provides "labels" which may be attached to objects and subjects (processes). The labels define different sensitivity levels and provide rules for access control. Use of MAC will be necessary if a system uses different sensitivity levels. It is not clear from the AIS whether or not a given system includes different sensitivity levels.

POSIX 1003.6 provides auditing functionality. The specification primarily defines an example interface which allows an application or function to generate audit records in a standard format. The specification does not describe how auditing is enabled for a user, file, or process.

POSIX 1003.6 provides access control via basic permissions and ACLs.

11.6 ID Administration And Ownership

11.6.1 Issuance

Interpretation

JSC procedure. For discussion of the different types of ID's, refer to section 11.6.4, which deals with the responsibilities of the different types of users.

Requirement

POSIX 1003.6 does not directly support the 5 types of user IDs. However, several of the types may be logically implemented with permissions and ACLs.

11.6.2 Annual Revalidation

Interpretation

JSC procedure. This procedure would be aided by automated support which retains the dates at which time each user ID was last utilized. This is a simple accounting procedure and does not add an appreciable amount of system overhead. This information would be useful in determining which accounts could be removed during the annual (or periodic) revalidation procedure.

Requirement

POSIX 1003.6 does not specify saving of any user-specific time stamp information to the file containing user IDs. If such automated record keeping is required, then it must be determined whether it belongs in POSIX 1003.6 or POSIX 1003.7.

11.6.3 Expiration

Interpretation

JSC procedure. This procedure will also benefit from automated accounting. Each user ID and password could have an associated creation time and an expiration period. When this period expires, a notification could take place or the system could automatically disallow subsequent access attempts. This mechanism would also allow the expiration to be set for different types of users (short expiration for privileged and high-sensitivity users and long for others).

Requirement

POSIX 1003.6 does not specify use of expiration times for user IDs and passwords. Again, if this function is required, it may be added to POSIX 1003.6 and POSIX 1003.7.

11.6.4 ID Owner Responsibilities

11.6.4.1 Personal ID Owner Responsibilities

Interpretation

Leaving a terminal session unattended is a problem and must be controlled with procedures. An electronic alternative is to require a "watchdog" daemon which logs the user off after a defined period of inactivity. Such daemons add overhead and often are forced to take unpleasant actions against users (such as terminating an edit session or a CPU bound process).

Protecting user data, keeping passwords secret, reporting expired IDs, and reporting disclosed passwords are controlled via procedures.

Requirement

POSIX 1003.6 provides permissions and ACLs for protecting of user data. It is however up to the discretion of the user and system administrator to use these mechanisms to protect data.

1.6.4.2 Privileged ID Responsibilities

Interpretation

The AIS implies that multiple privileged user IDs are required. This is necessary to allow individual accountability, as if all privileges were through a "root" ID, it would not be possible to track the individual responsible for a security violation.

All actions taken with a privileged account should be audited automatically by the system. The audit process must be real-time and be assured to complete before the privileged action is allowed to take place. A reasonable procedure will be to utilize a specially protected write-only device (printer or disk) to which audit information is output.

Privileged ID passwords should expire more frequently than normal user IDs. It may also be necessary to review IDs more often than annually.

Providing "templates" of privileges for different tasks would be useful, but difficult to implement. It may be more appropriate to designate such users as non-privileged and assign ownership of the appropriate files to allow access.

Privilege granting is a JSC procedure.

If additional environmental controls are required, it would be possible to implement a system which only allows privileged access during periods of time or only when a designated system administrator is currently logged on to the system.

Requirement

POSIX 1003.6 does not yet define privileged access. Although the scheme for privileged access is not known, it is suspected to be similar to the UNIX usage of the "root" ID. This is a problem as there is no way to account for individual actions, as all privileged users would use the same ID. A solution is to require an interface in 1003.6 which specifies creation of individual IDs, each with the equivalent of root access. Another solution is to disallow direct root access. In order to use root, a user must first log in as a normal user and then change to root.

POSIX 1003.6 does not specify the relationship between privileged use and protection of security related files. In particular, will it be possible for a privileged user to circumvent the security system (by updating or disabling the audit system)? This is probably possible, but should be noticed with real-time audit record generation.

11.6.4.3 Project ID Responsibilities

Interpretation

It appears that a project ID is the same as a personal ID, except that the owner is responsible for insuring that the project data is not accessed outside of the project group. This may be performed by setting the permissions and ACLs on data objects so that project users

alone can access the data as required (reading, writing, and executing). The project ID user could also allow new users to access the project data by updating the group or changing ACLs.

Reviewing "all access lists" most likely refers to determining if all users still need access to data (still belong in the group) and if all permissions are correct.

Requirement

POSIX 1003.6 provides all functionality necessary to support logical project IDs. A project would most likely correspond to a POSIX 1003.1 group.

11.6.1.4 System Service IDs

The distinction between system service ID and a privileged ID is not clearly defined. Does a system service user require privileged access? A user responsible for maintenance of system software will normally require privileged access, especially if the intended definition of "system" includes operating system, security, graphics, or other software.

It appears that the system service ID may be interpreted as a special form of privileged user which is temporarily provided to allow vendors and support personnel to periodically update the system software. The statement "the ID Manager may grant authorization to login to the ID to other users" implies that this ID will be temporarily provided to users who must perform system maintenance. This of course could cause problems if the password is not immediately modified after the ID is used.

All activity on a system service ID should be audited as described for a privileged user ID. Passwords should be treated in the same manner as privileged IDs, rather than in the same manner as personal IDs.

An alternative approach is to set the permissions of all files pertaining to a "system" to be owned by the system service ID user. In this way, that user could change the software as needed. Again, this is not foolproof, as unless all objects are isolated, it would be a problem to add new objects or directories.

Requirement

POSIX 1003.6 does not directly specify this type of user ID. As indicated, it may be implemented via a privileged ID or a user ID which owns all relevant objects. It may be most appropriate to remove this ID type and classify it as privileged.

11.6.4.5 Generic ID Responsibilities

Interpretation

The distinction between generic and personal IDs is not clear. It does not appear that generic IDs are "guest" accounts, as this is a basic violation of a secure system. Generic IDs also do not appear to be common IDs which are shared by multiple users, as this would prevent individual accountability. Rather, generic IDs are similar to personal IDs, except that a particular users access is fairly short-lived (such as a training exercise in which a temporary ID is assigned for use).

It is a necessary procedure to immediately change the ID's password once a user has completed its use. The expiration time on such ID passwords should be very short.

Requirement

POSIX 1003.6 provides the functionality described (assuming the interpretation described).

11.6.5 ID Management

Interpretation

JSC procedures.

Requirement

N/A.

11.6.6 Non-trivial Passwords

Interpretation

Most of the constraints described are reasonable and should be implemented as JSC procedures. This is primarily due to the fact that whether or not a password is "trivial", is a subjective decision which cannot be automated.

An additional constraint is a procedure which insures that users do not use the same password for multiple systems. This is a serious problem when users are active on systems at different sensitivity levels. On non-secure development systems, users often allow their passwords to be known by other users. If the same password is used on a secure system, it would be possible for an unauthorized user to gain access.

When a new system is received, it is a sound procedure to insure that all IDs (especially privileged IDs) are updated to use valid non-trivial passwords. When shipped, most systems will only include a few IDs necessary for system operation. This includes a privileged ID which will not be password protected (UNIX root).

Requirement

POSIX 1003.6 does not describe use of passwords. This is also absent from POSIX 1003.1. If any of the constraints described are to be automated, modification must be made to the appropriate specification. It would be reasonable for the system to implement the following constraints:

- Password not equal to user ID.
- Password does not consist of obvious strings (the strings are listed in a file which may be updated by system administrator).
- Updated passwords must not be successive - during password reviews, the system must insure that changed passwords are actually different. It is also necessary to prevent a user from changing to a temporary password and then back to the previously used password.

Access and maintenance of passwords may be most appropriately described in POSIX 1003.7

11.9 System Security

User Identification

Interpretation

JSC procedure.

Requirement

N/A.

Application Independence

Interpretation

This capability will be provided via basic owner, group, and other permissions for read, write, and execute. ACLs may be used if necessary.

Requirement

POSIX 1003.1 provides the owner, group, and other permissions for read, write, and execute. POSIX 1003.6 provides additional access control via ACLs.

Maintenance of the Security System

Interpretation

The definition of "real-time" is unclear? Is this true real-time response or is it simply "on-line" in which the security record updates are not queued? Queueing records may allow an individual to intercept and remove them before they are logged. If the covert action is to disable the system, then if the record is not logged first, the system will fail before the violation is recorded. This case demonstrates that true real-time logging is required (at least in special cases).

Access controls include basic permissions and access control lists. Operator capabilities will be limited via these access controls. Password changes will be audited and only allowed for privileged users (and possibly for a user's own ID).

It is assumed that "system programmers" are privileged users. There must be a mechanism which prevents privileged users from affecting audit files (at least without the knowledge of the system).

Requirement

POSIX 1003.6 does not specify whether or not audit record updates are logged in "real-time". This is a reasonable requirement.

POSIX 1003.6 defines the basic access controls and ACLs. The specification does not describe any means of controlling access to passwords.

POSIX 1003.6 does not specify how audit files are protected from privileged users.

Security System Administration and Monitoring

Interpretation

JSC procedures. "Self-auditing" is assumed to involve a privileged user having access or control of audit information generated for himself. Self-auditing could be prevented by disallowing access to security files, real-time audit record logging, or logging of all privileged actions to a write-only device.

Requirement

POSIX 1003.6 does not specify the manner in which self-auditing is prevented.

Auditability

Interpretation

Security auditing must be provided and it must be possible to enable/disable auditing for the following:

- User ID (including privileged IDs)
- Process
- File (such as the password file)

The system must allow audit for a controlled period of time. The system must also provide a set of functions which allow the audit records to be reviewed in a straight-forward manner.

Requirement

POSIX 1003.6 defines the basic functions required to generate audit events. These functions may be included in a trusted application to generate audit records. POSIX 1003.6 does not specify the system utilities which allow control of the audit system for user IDs and objects. This is apparently to be covered in the POSIX 1003.7 specification or a later draft of POSIX 1003.6.

POSIX 1003.6 does not specify any functions which allow review of the audit records. The specification defines a standard format which will allow other functions to review the audit data. Such review functions/commands will probably be found in POSIX 1003.7.

Terminal Operator Training

JSC procedure.

Requirement

N/A.

11.9.6 Password Usage

11.9.6.1 Composition

Interpretation

An automated password system is required to verify that the generated or selected password is composed of characters from a subset of at least 10 characters taken from the set of 95 graphics characters. It is unclear whether this automated password system is generating passwords for users and how the subset of 10 characters is selected.

Requirement

POSIX 1003.6 does not provide this function.

11.9.6.2 Length Range

Interpretation

This section calls for the password to be at least six characters long and to allow for a minimum of 10,000 possible passwords when used in conjunction with the number of charac-

ters in the composition subset. An automated password system shall verify that only passwords having a length within the acceptable length range shall be generated or selected whenever a password is created or changed. The selected password length will be an indication of the level of privilege associated with the password.

Requirement

POSIX 1003.6 does not provide this function.

11.9.6.3 Lifetime

Interpretation

Passwords shall have a maximum lifetime of 6 months.

Passwords which are suspected of being compromised should be replaced within one working day from the time of suspicion. Passwords should be deleted or replaced within three working days from the time that an owner is no longer an authorized system user, or any one of a set of owners is no longer authorized access to the data.

Passwords forgotten by their owner shall be replaced, not reissued. This is a procedure.

An automated password system shall allow the ID Administrator to delete or replace a password and will be capable of maintaining a record of the password transaction.

Requirement

There is no reference in the POSIX 1003.1 or the 1003.6 document to a function which will allow update of a password. There are functions for retrieving, but not for setting.

The only reasonable approach to monitoring the life of a password is to store the date of the last change, calculate an expiration date based on the time of creation or change, and then store the expiration date also. An automated routine could then be used to periodically check for passwords which have expired. The storage of these dates could be useful for other security requirements as well.

11.9.6.4 Source

Interpretation

All passwords must be randomly selected from a list of acceptable passwords or a string which is not at all related to a user's personal history. This is a JSC procedure.

Passwords selected or created shall be tested by the automated password system to assure that they meet the specifications of composition and length established for the AIS system before they are accepted as valid passwords.

Requirement

POSIX 1003.6 does not provide these functions.

11.9.6.7 Ownership

Interpretation

JSC procedures.

Requirement

N/A.

11.9.6.6 Distribution

Interpretation

Passwords are created and assigned to users by the automated password system. Any time a change is made to the password, an audit record containing the date and time of the password change and the identifier associated with the password is created and made available to authorized security personnel.

When the passwords are distributed from the password source, the temporary storage of the password should be erased, and the permanent storage of the password should only be accessible to the owner and the protected password system. There is no mention of privileged users being able to access the passwords.

Requirement

POSIX 1003.1 has deleted the storage of encoded passwords from both the *passwd* and *group databases*. The standard provides functions for keyed lookup of passwords so that it may not be possible for an application to browse the system databases indiscriminately. Where the passwords are stored is not clear.

11.9.6.7 Storage

Interpretation

Stored passwords will be protected in such a way that only the password system is authorized access to a password. Passwords that are encrypted before they are stored shall be protected from direct substitution.

Requirement

POSIX 1003.6 does not provide this function. Passwords are encrypted before they are stored and are never decrypted. Whenever an entered password needs to be verified, the entered password is encrypted and then compared to the stored encryption. This helps to protect the access to the password.

11.9.6.8 Entry

Interpretation

The system will not echo a password when being entered by the user. It must not be possible for the user (or another user) to modify this behavior.

It must not be possible to emulate the login process on a terminal in order to "steal" passwords from a user attempting system access.

The number of allowed password entry attempts (retries after incorrect password entry) shall be limited to a number selected by the system administrator. An ID shall be locked out in response to exceeding the maximum number of retries specified by the system administrator.

Invalid logins to a user ID shall be tracked and reported by the security system. Notification of invalid access attempts should be made to the security administrator and the user.

Requirement

POSIX 1003.6 does not provide any of the listed functions.

11.9.6.9 Transmission

Interpretation

It is not clear what this requirement is trying to address. The entered password will be encrypted (if necessary) and then compared to the stored password. All movement and comparisons will occur in the address space of a process. It is not likely that this data space will be violated.

Passwords transmitted between the place of entry and the place of comparison with the stored password shall be encrypted at the place of entry if the data that the password is protecting is encrypted at the place of data entry.

Requirement

Since it is not clear what this requirement is addressing, it is unknown what the requirement in POSIX may be.

11.9.6.10 Authentication Period

Interpretation

Personal passwords shall be authenticated each time a claim of identity is made, e.g., when "logging in to" an interactive system.

Access passwords shall be authenticated during the initial request for access to protected data.

Requirement

There is no mention of password authentication in the POSIX 1003.6, however most UNIX systems will request a password whenever access is attempted.

13. AIS Security Infractions Violations

13.2 Detection

13.2.1 System Resource Monitoring

Interpretation

The basic permissions and ACLs may be used to control access to most system resources (such as files, devices, and system applications). Abuse will be prevented by not allowing access to these resources. Abuse of resources such as CPUs, printers, and storage devices is prevented by use of quotas which limit the amount of access, time, or space allocated to a given user.

Requirement

POSIX 1003.6 does not include any mechanism for controlling resource allocation. Many UNIX systems provide an accounting/quota mechanism which may be used for this purpose. This however is more appropriately placed in POSIX 1003.7.

13.2.2 Audit Logs

Interpretation

All actions taken by privileged users should be audited. While this may not be the constant operating state, it must be possible to enable or disable this operation.

Requirement

POSIX 1003.6 provides the basis for auditing. As previously described, the POSIX 1003.6 specification does not address the manner in which auditing is enabled for a given user or object (file or application). This however is a valid requirement and must be present in the POSIX 1003.6 or 1003.7 specifications.

As previously described, it is not clear how POSIX will prevent privileged users from accessing audit logs.

13.2.3 Exception Reports

Interpretation

Generation of exception reports is useful for picking out the audit events which pertain to a suspected security violation.

Requirement

POSIX 1003.6 does not specify any functions or commands which may be used for audit log post-processing. POSIX 1003.6 does however specify a standard format for audit records which will simplify development of functions and commands which provide post-processing. Such functions will most likely be described in POSIX 1003.7.

13.2.4 Operator Logs

Interpretation

Operator logs would be a useful way of monitoring system activity. Real-time generation and review of audit records will prevent even a privileged user from adversely affecting the security system. In the absence of a human operator, the audit records could be output to a printer or other write-only device.

Requirement

POSIX 1003.6 does not specify the manner in which audit records are generated. Whether or not the records are generated in real-time and the ability to route to a specific destination appear to be implementation-defined. Generation of real-time audit records is a valid requirement.